AD-A095 372 YALE UNIV NEW HAVEN CT DEPT OF COMPUTER SCIENCE F/G 9/2
CONCEPTUAL INFORMATION RETRIEVAL.(U)
DEC 80 R C SCHANK, J KOLODNER, G DEJONG N00014-75-C-111
UNCLASSIFIED RR-190 NL

1 OF 1
AD-
A095372

END
DATE
FILMED
3-81
DTIC

Conceptual Information Retrieval

Roger Schank, Janet Kolodner, Gerald DeJong

Research Report #190

December 1980

YALE UNIVERSITY
DEPARTMENT OF COMPUTER SCIENCE

Conceptual Information Retrieval

Roger Schank, Janet Kolodner, Gerald DeJong

Research Report #190

December 1980

This work was supported in part by the Advanced Research Projects Agency of the Department of Defense and monitored under the Office of Naval Research under contract N00014-75-C-1111.

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS
BEFORE COMPLETING FORM

1. REPORT NUMBER
#190 ✓

2. GOVT ACCESSION NO.
AD-A095

3. RECIPIENT'S CATALOG NUMBER
372

4. TITLE *(and Subtitle)*
(6) Conceptual Information Retrieval,

5. TYPE OF REPORT & PERIOD COVERED
(9) Technical Report,

6. ~~PERFORMING ORG. REPORT NUMBER~~

7. AUTHOR(s)
(10) Roger C. Schank, Janet L. Kolodner, Gerald DeJong

8. CONTRACT OR GRANT NUMBER(s)
(15) N00014-75-C-1111

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Yale University - Department of Computer Science
10 Hillhouse Avenue
New Haven, Connecticut 06520

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
(12)

11. CONTROLLING OFFICE NAME AND ADDRESS
Advanced Research Projects Agency
1400 Wilson Boulevard
Arlington, Virginia 22209
(11)

12. REPORT DATE
December 1980

13. NUMBER OF PAGES
45

14. MONITORING AGENCY NAME & ADDRESS *(if different from Controlling Office)*
Office of Naval Research
Information Systems Program
Arlington, Virginia 22217
(14)

15. SECURITY CLASS. *(of this report)*
unclassified

15a. DECLASSIFICATION DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*
Distribution of this report is unlimited.

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

Intelligent information retrieval
Data base management
Natural language processing
Cognitive modeling
Memory organization
Artificial Intelligence
Heuristic Searching
Integrated Understanding
Automatic data base updating

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

If we want to build intelligent information retrieval systems, we will have to give them the capabilities of understanding Natural Language, automatically organizing and reorganizing their memories, and using intelligent heuristics for searching their memories. These systems will have to analyze and understand both new text and Natural Language queries.

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

In answering questions, they will have to direct memory search to reasonable places. This requires good organization of both the conceptual content of texts and knowledge necessary for understanding those texts and accessing memory.

The CYRUS and FRUMP systems (Kolodner (1978), Schank and Kolodner (1979), DeJong (1979)) comprise an information retrieval system called CyFr. Together, they have the analysis and retrieval capabilities mentioned above. FRUMP analysis news stories from the UPI wire for their conceptual content, and produces summaries of those stories. It sends summaries of stories about important people to CYRUS. CYRUS automatically adds those stories to its memory, and can then retrieve that information to answer questions posed to it in natural language.

This paper describes the problems involved in building such an intelligent system. It proposes solutions to some of those problems bases on recent research in Artificial Intelligence and Natural Language Processing, and describes the CyFr system, which implements those solutions. The solutions we propose and implement are based on a model of human understanding and memory retrieval..

-- OFFICIAL DISTIRUBTION LIST --

12 copies

Defense Documentation Center
Cameron Station
Alexandria, Virginia 22314

2 copies

Office of Naval Research
Information Systems Program
Code 437
Arlington, Virginia 22217

3 copies

Dr. Judith Daly
Advanced Research Projects Agency
Cybernetics Technology Office
1400 Wilson Boulevard
Arlington, Virginia 22209

1 copy

Office of Naval Research
Branch Office - Boston
495 Summer Street
Boston, Massachusetts 02210

1 copy

Office of Naval Research
Branch Office - Chicago
536 South Clark Street
Chicago, Illinois 60615

1 copy

Office of Naval Research
Branch Office - Pasadena
1030 East Green Street
Pasadena, California 91106

1 copy

Mr. Steven Wong
New York Area Office
715 Broadway - 5th Floor
New York, New York 10003

6 copies

Naval Research Laboratory
Technical Information Division
Code 2627
Washington, D.C. 20375

1 copy

Dr. A.L. Slafkosky
Commandant of the Marine Corps
Code RD-1
Washington, D.C. 20380

1 copy

Office of Naval Research
Code 455
Arlington, Virginia 22217

Office of Naval Research — 1 copy
Code 458
Arlington, Virginia 22217

Naval Electronics Laboratory Center — 1 copy
Advanced Software Technology Division
Code 5200
San Diego, California 92152

Mr. E.H. Gleissner — 1 copy
Naval Ship Research and Development
Computation and Mathematics Department
Bethesda, Maryland 20084

Captain Grace M. Hopper — 1 copy
NAICOM/MIS Planning Board
Office of the Chief of Naval Operations
Washington, D.C. 20350

Dr. Robert Engelmore — 2 copies
Advanced Research Project Agency
Information Processing Techniques
1400 Wilson Boulevard
Arlington, Virginia 22209

Professor Omar Wing — 1 copy
Columbia University in the City of New York
Department of Electrical Engineering and
Computer Science
New York, New York 10027

Office of Naval Research — 1 copy
Assistant Chief for Technology
Code 200
Arlington, Virginia 22217

Major J.P. Pennell — 1 copy
Headquarters, Marine Corp.
(Attn: Code CCA-40)
Washington, D.C. 20380

Computer Systems Management, Inc. — 5 copies
1300 Wilson Boulevard, Suite 102
Arlington, Virginia 22209

Ms. Robin Dillard — 1 copy
Naval Ocean Systems Center
C2 Information Processing Branch (Code 8242)
271 Catalina Boulevard
San Diego, California 92152

Conceptual Information Retrieval

Roger C. Schank, Janet L. Kolodner, and Gerald DeJong
Yale University
Dept. of Computer Science
Box 2158 Yale Station
New Haven, CT 06520 USA

ABSTRACT

If we want to build intelligent information retrieval systems, we will have to give them the capabilities of understanding Natural Language, automatically organizing and reorganizing their memories, and using intelligent heuristics for searching their memories. These systems will have to analyze and understand both new text and Natural Language queries. In answering questions, they will have to direct memory search to reasonable places. This requires good organization of both the conceptual content of texts and knowledge necessary for understanding those texts and accessing memory.

The CYRUS and FRUMP systems (Kolodner (1978), Schank and Kolodner (1979), DeJong (1979)) comprise an information retrieval system called CyFr. Together, they have the analysis and retrieval capabilities mentioned above. FRUMP analyzes news stories from the UPI wire for their conceptual content, and produces summaries of those stories. It sends summaries of stories about important people to CYRUS. CYRUS automatically adds those stories to its memory, and can then retrieve that information to answer questions posed to it in natural language.

This paper describes the problems involved in building such an intelligent system. It proposes solutions to some of those problems based on recent research in Artificial Intelligence and Natural Language Processing, and describes the CyFr system, which implements those solutions. The solutions we propose and implement are based on a model of human understanding and memory retrieval.

Conceptual Information Retrieval

Roger C. Schank, Janet L. Kolodner, and Gerald DeJong

1.0 Introduction

In a sense, all people who work with computers have awaited the day when computers would finally know enough so they wouldn't be so difficult to deal with. The field of Artificial Intelligence has been devoted to the attempt to make computers "smart". What would a smart computer contribute to the field of Information Retrieval? Or, to put the question another way, what do we want the ideal information retrieval system to be capable of doing? The simplest answer to this question is that, ideally, a computer should really know about what it is looking for when it searches its memory. We would like our retrieval programs to come up with information that was not exactly what was asked for, but nevertheless fits the bill. We would like partial matches, where half an answer is better than none. We would like to be able to talk to our program in natural English. And, perhaps most importantly, we would like our program to understand what we are after well enough for it to be capable of giving us what we really wanted, rather than what we actually asked for. How far away are we from all this?

Recent research at the Yale Artificial Intelligence Project has been directed towards finding out how to do some of the tasks mentioned above. The question we have posed for ourselves is, can we design a system that knows a great deal of information about a subject that we can query in a natural way? This is a rather different question than is usually asked in Information Retrieval research. We are not starting out with a data base of information that we would like to query. Rather, we are engaged in a theoretical enterprise that asks how a data base ought best be structured in order to facilitate communication with it.

It follows then, that we are interested in organizing the meaning of the data we are dealing with rather than the raw data itself. This sets our work apart from the more garden variety Information Retrieval work in which information is stored independent of meaning.

The reason to organize the meaning of one´s data in the data base is simply that the meaning of that data is what one wants to find out about. If the meaning of the data is available, then English questions that relate to that meaning can be entertained. Certainly the idea of accessing a data base in English is not new. Why hasn´t it been seriously attempted before?

Most research in Information Retrieval has been concentrated on document retrieval and not on problems of organizing and retrieving the information contained in those documents. Thus, most IR systems require the user to pose his questions as a complex boolean or extended boolean expression of key words for specifying documents. He must then sort through the retrieved documents himself for the information he needs. Many times the required boolean expression is so complex that only a technician with intimate knowledge about the system, i.e., somebody familiar with both its content and idiosyncrasies, can formulate that expression. These things make it extremely hard for a novice or casual user to access the system, or for even an experienced user to get the information he needs quickly. Clearly it would be advantageous to eliminate these problems by the use of natural English. But, the natural language problem being very hard, many computer workers have chosen to get around the problem by the use of key word schemes.

Thus data bases tend to use information that is indexed lexically by key words found in the title, abstract, or author´s key word list. Besides making retrieval from the data base difficult for users, key words are rather poor as an organizational tool. Key words are not sufficient for expressing the conceptual content or meaning of the contents of a document. In a system which relies on key words and their synonyms, relevant documents in the data base will often be missed because they do not contain specified words, and many unrelated documents might be retrieved. For example, searching any news data base for stories involving ex-United States President Gerald Ford would yield many stories about the Ford Motor Car Company. The problem is that while the two Fords are conceptually very different, they are lexically identical.

These problems exist even in systems with automatic synonym lists or thesaurus capabilities. In addition, this type of organization does not provide for indexing individual items or facts contained in documents. Because information from the documents is not extracted and stored, the real questions people have about the contents of documents cannot be answered by the systems.

One way IR researchers have been attempting to deal with key word and content analysis problems is by putting texts into a normalized form (Allen, 1966; Grishman & Hirschman, 1978) and then indexing and storing the normalized form instead of the natural language text itself. This approach, however, still has problems. In Allen's legal rules system, the analysis of rules into normalized form must be done manually. While Grishman et al. use syntactic rules for automatically analyzing the contents of medical records and producing a normalized form, they admit that the records they have analyzed form a "specialized subset of English" in which the vocabulary demands are relatively small and the syntax is rather stylized. Thus, they seem to achieve success, but their results are not easily extendible to less constrained domains.

In AI, there have been two approaches to these problems. One approach has been the development of natural language front ends for data base and information retrieval systems. Some of those systems are LUNAR (Woods, et al., 1972), PLANES (Waltz, 1978), and LIFER (Hendrix, et al., 1978). Although these researchers have developed front end interfaces for dealing with people's natural language queries, none have addressed the problems of organizing the data in the system. They have generally taken data bases that were already around, and have not gone into the theoretical issues of how the knowledge and information necessary for understanding should best be organized.

The second approach, which we have been pursuing, has been to deal with the interaction of prior knowledge and understanding, and has been addressing issues of representation and organization of that knowledge. Because people manipulate language so well, we have tried to find out how people use knowledge in understanding, and then to model those processes on the computer. Our research in Natural Language Processing has now reached a point where it can and should be useful in developin smarter IR systems. Many of the natural language and memory organizacion problems we have been dealing with are problems that must be addressed in building such a system. We have been developing conceptual primitives (Schank, 1975; Schank and Carbonell, 1979) to use in representing meanings, and have systems (ELI (Riesbeck & Schank, 1976) and CA (Birnbaum & Selfridge, 1979)) which parse or analyze English sentences into a representation of their meaning based on those primitives. Certainly, a natural language query system needs to be able to extract the meaning from questions it is asked.

We have also been concerned with story understanding. The SAM system (Cullingford, 1978), for example, read simple stories and used its knowledge about stereotypic situations called scripts (Schank and Abelson, 1977) to infer information that was only implicitly in the story. On reading that John went to a restaurant and ordered lobster, it was able to infer that John also ate the lobster. It could do that because its knowledge about restaurants told it that people usually eat what they order. The PAM system (Wilensky, 1978) was a system designed to understand goal-related stories. If it read that John needed money, got a gun and went to a liquor store, it would predict that John had the goal of robbing the liquor store to get the money he needed. Finally, the FRUMP (DeJong, 1979) system implements a theory of skimming, and extracts summaries from newspaper stories in a number of knowledge domains. If an information retrieval system is going to extract information from its documents, it needs these capabilities.

One other major concern we have had related to IR is that of organizing large quantities of information in memories (Schank, 1979; Kolodner, 1978). The CYRUS system (Kolodner, 1978, 1980) was developed as a long term memory for information about people. We began by asking questions such as "how can we organize all of the events in a person´s life so that they can be retrieved when needed?" Our first answer to that question involved breaking up a person´s time line into manageable chunks called eras (Kolodner, 1978), and storing events in those structures. Although eras were useful, they were insufficient, and our representational and organizational schemes have now evolved into a general organizational scheme for events and a theory of Memory Organization Packets (MOPs) to organize events and event information in memory. CYRUS´ memory is now based on MOPs. We have also been developing another story understanding system IPP (Lebowitz, 1979) whose processing is based on MOPs. At this point, our understanding of the kinds of structures necessary for representing and organizing large amounts of information and knowledge in the computer can be useful in building the ideal IR system.

Two of our systems -- CYRUS and FRUMP -- have been connected together to form the beginning of such an IR system -- CyFr. FRUMP reads and summarizes stories from the UPI wire. It produces conceptual representations of the events in those stories, and when it reads a story about one of the important people CYRUS is tracking, it sends that summary along to CYRUS. CYRUS has been storing information about former U. S. Secretary of State Cyrus Vance, and now that he has resigned will go on collecting stories about his successor, Edmund Muskie. When CYRUS receives a story about one of these men, it adds the information to its memory, automatically creating new organizational categories in its memory as needed. Later, CYRUS can be queried in natural language about those new events. The following is a sample run of the entire system on a recent story about Muskie:

```
*****************

@RU FRUMP

Dictionary loaded
Building script trees
Dictionary fork...Done

*(SKIM (MAY130.K05]

FILE (MAY130 . K05) SKIMMED AT 5:31PM ON 5-14-1980

INPUT:  a074 r i ss czc bsa u v
     -Skie 1 d-picp6thraf 5-13-------------- (Me lves foreure (
     ---By Jim Anderson---Washington (Upi)-Secretary of State Edmund
Muskie flew to Europe on his maiden diplomatic mission today to whip
up allied support for U.  S.  policies toward Iran and to meet with
veteran Soviet Foreign Minister Andrei Gromyko.
     Muskie met for 20 minutes with President carter, then left nearby
andrews Air Force Base for Brussels shortly after 9 a.  m.  Edt.
     Muskie will meet with NATO foreign ministers in Brussels
Wednesday, then go to Vienna for a meeting with Gromyko at the 25 th
anniversary celebration of the Austrian State Treaty.
     Monday, Muskie told a group of business executives:  "There has
been a perception that the allies are less than enthusiastic" about
the program of U.  S.  economic sanctions against Iran.  mandkow i am
ous abutt, "esai.  pcup6ta:  kesd Ao------------------5-13 9:41
a----kkk

SELECTED SKETCHY SCRIPT $MEET

CPU TIME FOR UNDERSTANDING = 87463 MILLISECONDS SKIM done

** Generating ...
```

```
ENGLISH:
     Secretary of State Edmund Muskie went on a plane from the United
States to Europe today.  Secretary of State Edmund Muskie had talks
with Carter during a briefing.  Secretary of State Edmund Muskie gave
a speech to a group on Monday.  Secretary of State Edmund Muskie will
have talks with NATO in Brussels on Wednesday.

sending summary to CYRUS -- file ((CYRUS FRUMP-DUMPS) (MAY130.R05))

******************

@cyrecv

*(PROCESS)

reading in file ((CYRUS FRUMP-DUMPS) (MAY130 . R05)) from FRUMP
converting file to CYRUS format ...

Conversion complete.
Sending file ((CYRUS FRUMP-DUMPS) (MAY130 . F05)) to CYRUS

******************

@CYRUS

today is Friday, May 16, 1980.

*(PROCESS-FILES)

reading in file (MAY130 . F05)
updating memory with new events ...
updating complete

ready to answer questions

>Where is Muskie today?
  ...searching memory for question concept
Probably in the United States.

>Where was Muskie Tuesday?
   ...searching memory for question concept
In Europe.

>Why did he go there?
   ...answering question using previous context
To meet with Gromyko.

>Who else did he meet with there?
   ...answering question using previous context:
With NATO in Brussels on Wednesday.

>When was the last time Muskie made a speech?
  ...answering question using time context
On Monday, May 12.
```

****

2.0 The issues

There seem to be four key issues to be addressed in designing a system to deal with the organization and retrieval of facts in relatively unconstrained domains (e.g., current events, biographical information, scientific abstracts). First, the system should be able to automatically understand natural language text -- both input to the data base and queries to the system. This understanding involves producing a representation of the content of the natural language input. Second, information in the system should be formatted and organized (automatically) in such a way that the conceptual content or meaning of an item can be used for retrieval rather than simply its key words. The representations produced by the parser should be adequate for this task, and should be automatically integrated into the memory organization. Third, the system needs rules for accessing its memory -- both for directing input of new information into the memory and for directing retrieval. Those access rules will need to use knowledge about the memory's domains of information. Thus, the fourth issue to be addressed is the extent and organization of that knowledge. Finally, rules for answering questions and generating good answers must be specified. The system should be based on a theory of content analysis, inference, and organization that can be extended to domains other than those for which the system was specifically designed. Some of the issues that theory will have to address follow:

1. automatic analysis of natural language text -- for both question answering and adding new data to the data base
2. organization of the content of that text according to

its conceptual content

3. memory access functions for direction of both input and retrieval
4. organization and extent of knowledge about the domain necessary for understanding, updating, and retrieval
5. Automatic updating of memory
6. Automatic creation of new categories
7. Automatic search key construction from natural language text
8. Strategies for searching the data base

These issues fall into four major categories:

I. organization of memory

II. understanding input text

III. automatically updating memory

IV. answering questions

Recent research in natural language processing has been addressing a number of these problems. In the next sections, each of these issues will be discussed briefly, and a computer system CyFr based on this approach will be presented.

3.0 Memory organization

Unlike traditional IR, which depends on key words to specify categories for documents, in a fact retrieval system, the organization of memory must be able to make the contents of documents accessible for retrieval and inference.

An important issue in designing a system to work in a relatively unconstrained domain is that of organization of the system´s memory. Memory should be organized so that any given fact can be extracted from the data base when appropriate. In addition, the organization must allow inferences to be drawn from information in the data base in order to retrieve information that is only implicitly there.

One of the major items of research in natural language processing within Artificial Intelligence has been the problem of inference. Often, the meaning or implication of a sentence, story or event is not evident on the surface, and inferences must be made for full understanding. A good memory organization will allow the correct inferences to be made when needed. Consider, for example, the following two events.

(1) Vance met briefly with Gromyko yesterday in Washington.

(2) Vance met briefly with his wife yesterday in Washington.

On the surface, these events look similar, except perhaps for the occupations of the participants. The purpose and probable global context of each, however, are different. The first, a diplomatic meeting, is probably part of negotiations. We would not want to infer, however, that the second has anything to do with negotiations. If memory is organized to reflect these differences, then the negotiations implicitly described by (1) can be inferred, while different purposes would be inferred for (2). We assume Vance's meeting with Gromyko is part of negotiations; so should a computer system keeping track of Vance.

As in document retrieval, this requires that similar items be placed in the same category in memory. Unlike document retrieval, however, the items in the categories will be aspects of the contents of documents rather than documents themselves. Also unlike traditional IR systems, the categories will not be specified by key words, but will have to be specified by concepts they include. If categories are to be specified by concepts, then the system must have knowledge about the concepts defining each category. It needs that information for a number of reasons -- to recognize which categories a new item fits into, to recognize which categories a question is referencing, to make inferences relating categories to each other, and to make inferences relating individual items to other items in the data base. These issues will be discussed in more detail in later sections.

Consider, for example, a system which stores biographical information about people, where the daily activities of a person are constantly updating the system's knowledge of that person. In such a system, it seems reasonable to organize the memory of the system around event categories. We could imagine, then, that for a statesman, the system would have categories such as "diplomatic meetings", "negotiations", "diplomatic trips", etc. It might also seem reasonable to have indices into these categories from each place and person in the data base. A meeting between Cyrus Vance and Menachim Begin in Israel, in a system that was tracking Cyrus Vance, would be stored in a "diplomatic meetings" category, as well as in categories "Vance activities", "Begin activities", "activities in Israel", "activities in the Mid East", etc. In fact, the number and actual content of these categories would depend on the size of the data base and the similarities and differences between its items. If the data base held only 10 meetings, then a "diplomatic meetings" category would be appropriate. If, however, the data base held 100 or 1000 diplomatic meetings, the category "diplomatic meetings" would no longer be efficient for retrieval, and new sub-categories indexed under "diplomatic meetings" would have to be constructed. Suppose, for example, that many different events in the data base happened in Israel. In that case, a meeting between Vance and Begin in Israel might be stored in a "meetings in Israel" category, a sub-category of either or both of the categories "activities in Israel" and "diplomatic meetings", indexed off of either or both of them.

4.0 Analyzing input text

Extracting the meaning of input events from raw natural language text is a major obstacle in building a self-updating conceptually organized retrieval system. Consider again a system which keeps track of news event information. At first, it may seem possible to organize incoming information on the basis of certain key words present in the input text. After all, one might expect events with similar meanings to be described with similar words. For example, events of one country invading another ought to be filled with words like "attack, invade, incursion ...". However, this is not necessarily the case. In a recent news service article, an Israeli incursion into Lebanon was reported as follows:

(3) Israeli troops crossed into Lebanon today, according to an army spokesman in Tel Aviv.

In this sentence, there are no "give-away" action words like "invade" or "incursion". Surely, the word "crossed" ought not be a key word of the category for hostile military acts. There is still the possibility that the word "troops" might be used as the key word for this event. However, troops can participate in many events other than hostile ones. Troops often appear, for example, in stories describing natural disasters, parades, military exercises. Troops also might be sent to a friendly country as a form of military aid. These situations must not be categorized as hostile military acts.

It is the meaning of the entire sentence, including knowledge about the relationship of the countries involved, and not the key words, which permit understanding and subsequent memory organization. Any categorizing system based on key words rather than meaning will have trouble with ambiguous English words, and most English words are ambiguous to some degree. Consider, for example, the following sentence:

(4) Troops of sorrowful Cardinals crossed themselves as they entered the conclave for the second time in less than two months.

Here "crossed" no longer means "were transported", and "troops" no longer refers to "military units".

Nor is sentence (3) peculiar in its lack of an adequate "give away" key word. The vast majority of newspaper articles do not use simple straight forward key words to describe events. There is a good reason for this. News articles are written to be interesting to human readers. They are not written to be easily classifiable by computer systems. Thus, if we want to use people´s written descriptions of events (e.g., news wire services) to automatically update our system data base, we cannot rely on key word methods of categorizing events.

Instead, we use the meaning of an English text to categorize it. We represent events within the system in terms of a "meaning language" rather than as English sentences. Events with similar meanings have similar representations.

How can the "meaning" of an event be represented inside a computer? This is an important issue ı Artificial Intelligence commonly known as "knowledge representation". There has been much work done on knowledge representation and there are many knowledge representation systems. For example, see Bobrow and Winograd (1977), Charniak (1977), Norman, Rumelhart et al. (1975), Wilks (1977). We use a representation scheme called "Conceptual Dependency" (or CD) (Schank, 1975). The job of the text analyzer is to map input events (specified in English) into meaning representations (specified in CD). The categorizing system will then be effectively isolated from the eccentricities of how events are reported.

The text analyzer in a working information retrieval system must be able to automatically expand its data base. To do this, a working system must employ a system that will not "fall apart" when it encounters new words or unfamiliar usages. That is, the system must be able to process any new event without human intervention. It must, of course, know the meanings of many English words, and be able to deal with the ambiguities which are rampant in English texts. In section 7.1 we will describe the actual system used to analyze the English input that is to be added to the conceptual data base.

5.0 Automatically updating the data base

After being understood and formatted, a new data item must be added to the data base. The system should be able to automatically add each item to the categories it belongs in.

In traditional IR systems, categories for new inputs are chosen by extracting the content words from the input and doing a statistical analysis to see which category has the most key words referring to it (Salton, 1968; Heaps, 1978). The new data is then put into that category. There are a number of major problems with this scheme. First, as was discussed above, statistical analysis of key words will not always result in a correct category being chosen. For example, although the sentence "Israeli troops crossed into Lebanon today" probably refers to an attack, we would not want the key words "crossed" or "crossed into" as specifiers of a category "attacks". Second, some items might be relevant to a number of categories and putting them in only one will decrease their chance of being retrievable. Third, aspects of the contents of the document relate to each other, and using this scheme there is no way to extract the contents and draw the necessary inferences between different aspects of the text.

There are three major problems to be addressed in discussing automatic data base update -- choosing memory categories for new items, inferring the relationship between the new item and other items already in the data base, and maintaining good data base organization through automatic creation of new categories from old ones.

5.1 Categorizing items

If categories are specified by concepts, and if the natural language analyzer parses text into a conceptual representation, then inferences can be made from the conceptual representations (or meanings) of new items to decide which categories they belong in. Suppose, for example, the category "diplomatic meeting" is defined as talks between diplomats of different countries. In order to recognize that an item belongs in that category, its action and the occupations and nationalities of its participants would be checked against the specifications for "diplomatic meetings". Each time there was a match that item would be entered into the "diplomatic meetings" category. In order to do that recognition, the system would need knowledge about what types of occupations quality as "diplomatic" and what sorts of activities qualify as "diplomatic meetings". An input text, for example, might say any of the following and quality as a "diplomatic meeting":

(5) Vance met with Gromyko yesterday
(6) Vance and Gromyko talked for 3 hours yesterday
(7) The meeting included Vance and Gromyko
(8) Vance and Gromyko discussed SALT over lunch

On the other hand, the following instances, though worded similarly, would not fall into that class of activities.

(9) Vance met with Scharansky´s wife briefly yesterday morning
(10) Vance and Carter talked for 3 hours yesterday
(11) The meeting included Vance and his office staff
(12) Vance and his wife discussed SALT over lunch

In fact, the only difference between the first and second set is that the first involved diplomats from different countries, while the second involves meetings between non-diplomats or person representing the same country. Thus, knowledge about each of the people involved including their occupation and nationality is needed to categorize a discussion as a "diplomatic meeting". With that knowledge, the system would be able to place events whose descriptions matched those of a "diplomatic meeting" into the "diplomatic meeting" category.

The conceptual representations produced by the parsing mechanism, then, need to be used by the updating procedures to decide which categories are appropriate. In fact, the action part of the conceptual parse of an item may already specify the category it belongs in. If, for example, the memory did not differentiate between different kinds of "talks", and if "talks" were a primitive activity, then only the action of the item would have to be checked before deciding which category to put it into.

The procedure for deciding which categories an item belongs in should not have to check each category in memory for membership. Rather, that search should be guided by the contents of the new item. Organizing categories around the conceptual primitive produced by the parser enables that. In adding a meeting between Secretary of State Cyrus Vance and Foreign Minister Andrei Gromyko, its action "meeting" or "talks" will specify a class of activities which it could fall into, thus limiting the number of categories to be checked. Further processing would show that Vance and Gromyko are both diplomats, a specification which will further limit the number of possible meeting types. Thus, the knowledge the system holds and the organization of categories with respect to each other can be used to limit memory search. We will see this again in discussing retrieval.

5.2 Maintaining data base consistency

In adding new information to the data base, the same event should not be added more than once, but its new reference should be merged with the reference already in memory. Suppose the system described above was informed yesterday that Carter and Vance met and Carter told Vance to go to the Mid East to negotiate Mid East peace. If today the system were told that Vance was in Israel meeting with Begin, it should be able to relate those two pieces of information. In particular, this requires the system to "know" that diplomatic meetings are normally part of negotiations, that employees normally do what their bosses tell them, and that Carter is Vance's boss. Using that information, it should connect today's meeting and its purpose with yesterday's meeting, and should infer that today's meeting is part of Mid East negotiations.

We stated earlier that the system needed knowledge in order to relate categories to each other and new items to old items already in the data base. It is obvious from this example that such knowledge is also needed to update the data base intelligently. New information must be integrated with previous information. Retrieval routines can then make use of that additional information. Information that is not present in a document is sometimes as important as the information which is explicitly there, and knowledge associated with the system's categories can help in inferring that information.

5.3 Maintaining effective data base organization

Because it is impossible beforehand to anticipate all entries to an IR system, the system should be able to maintain good organization by automatically re-organizing its categories. When a category gets large, the system should be able to create new sub-categories from its specifications and contents. In traditional key word IR systems, new categories are created by doing a statistical analysis of key words in each of the entries, and dividing the category on that basis. This has all the drawbacks mentioned above.

Following the conceptual organization scheme and analysis we have been explaining, categories can be sub-divided according to their most appropriate conceptual or content differences. Consider again the system described above. Suppose, for example, it had a category "diplomatic meetings" into which it entered each meeting it read about. In designing the data base, the designers may have thought that "diplomatic meetings" would be an appropriately-sized category. However, as more information is added to the data base, the category "diplomatic meetings" may get out of hand. If the system keeps track of the similarities and differences between the items it has placed in that category (i.e., indexes the similarities and differences as it goes along), then it can effectively dividing the category into sub-categories.

Suppose, for example, that almost all meetings involved discussion of some international contract. In that case, a sub-category of meetings in which international contracts were discussed would be a bad sub-category of meetings. However, a category of meetings in which something other than an international contract was discussed would be a good sub-category. Thus, norms are not good category specifiers, but deviations from a norm are good. In addition, each domain might have aspects that are important to it. The destination of a trip, for example, is important, as is the topic of a meeting. Those important components can be used as a basis for splitting the category into sub-categories. Categories should serve to break the data base into workable units. They should not be so diverse that they have only one or a few members, but not so large that they contain almost all members of their super-category.

The point then, is that a good IR system must have the ability to change its categorization system as necessary. Now, for a data base that is organized by indices that do not express meanings, this may seem like a goal that beyond hope. And perhaps it is for such systems. But, when the basic data is meaning, and the memory organization used is dependent on a program that attempts to generalize meaning units in an attempt to form new ones, such a system is within the bounds of plausibility.

We achieve this aim in our CyFr system by storing each event in a number of categories. Principles of automatic generalization are then used to sub-divide a initial categories according to commonalities in future inputs. From the "diplomatic meetings" category, for example, relevant new categories might be "meetings about something other than an international contract", "meetings involving heads of state", "meetings that are not part of negotiations", "meetings in the Mid East", "meetings in Europe", etc. This sub-categorization cannot be done on a purely statistical basis. Categories should be efficiently sized, but the splitting of a particular category should be guided by (1) knowledge about what is important about that category and (2) the category´s norms. If the activity "diplomatic meetings" is known to be an occupational and political activity, then we would expect the system to keep track of occupational and political similarities and differences, and to use those as a basis for deciding how the category should be split.

6.0 Answering questions

An information retrieval system should understand questions that people pose to it in the language most convenient for them to use -- the one they speak. People should be able to formulate questions in natural language, and the system should be able to analyze the questions, extract their meanings, automatically create search keys from the natural language input, and retrieve and generate appropriate answers. This understanding process requires inference, a large store of knowledge about the domains to be understood, and an understanding of the structure of the data base -- the same kinds of knowledge necessary for categorization and organization maintenance. The key problems we will address in discussing retrieval are construction of search keys and strategies for search. Questions must be analyzed to produce a meaning representation for the same reasons described previously for input text (section 4.0).

If memory organizes its items according to their differences, then retrieval keys must specify unique properties of items in order to get distinct items from the data base. Because this will not always be possible, searching for related items which might refer to the items being sought will sometimes be necessary in searching memory. To find meetings between Vance and Begin, for example, which happen quite often, it might be necessary to retrieve negotiating episodes between the United States and Israel. Those episodes might include meetings between Vance and Begin, in which case meetings would be found. Even if the particular negotiations episodes found do not contain meetings between Vance and Begin, they might suggest topics for meetings which could be added to the initial specification of a meeting to create a better search key. Thus, search key construction and elaboration and guidance of search are important problems to address.

6.1 Automatic construction of search keys

Consider, again, the system described above which stores biographical information about important political people and has that information organized in event categories. Suppose this system were asked the question "What heads of state has Vance met?" To answer the question, it would have to decide which categories of memory would be appropriate to search. Using the same information described above to place events in categories, it would infer that the most likely place for Vance to meet these people is in diplomatic meetings. After making that decision, it could use its knowledge about the structure of memory (i.e., the types of categories it has) to create an appropriate search key. In this case, that search key would be a specification to look for "diplomatic meetings with heads of state". If a satisfactory answer were not found that way, the system would have to go on to decide where else in memory it might be appropriate to look. Because "diplomatic meetings" are normally part of negotiations, it might create and search for "negotiations episodes involving Vance and a head of state". Tnus, one important use of inference in a question answering system is construction of search keys, or deciding what to look for in the data base. That decision must not be done blindly, but must be directed to insure efficient search.

6.2 Necessary background knowledge

In order for a system to create search keys and determine which categories of memory it should look in, it has to have knowledge about the particular domains it will be searching for. In order to infer that the category "diplomatic meetings with heads of state" East" is appropriate for answering "What heads of state has Vance met?", it must know that heads of state are diplomats and normally converse at diplomatic meetings. If it were asked "Has Vance ever met Mrs. Begin?", we would not want it to search for a diplomatic meeting. Similarly, to infer that "diplomatic meetings in the Middle East" is appropriate for answering "Who has Vance negotiation Middle East peace with?", it must know the relationship between "negotiations" and "diplomatic meetings" and between the topic of an international contract and the place where activities involving it normally take place.

In discussing memory organization, we stated that an intelligent system must have knowledge about its categories. Suppose a data base has the category "meetings with heads of state." Each meeting it hears about with a head of state as a participant will go into that category. In order to search memory, it will also need information about when it is appropriate to search that category. It will be appropriate to search the category "meetings with heads of state", for example, not only for meetings between heads of state, but also when searching for such things as the start of negotiations, topics that heads of state are interested in, summit conferences, etc. That category should probably not be searched, however, when attempting to retrieve recent speeches Vance has given or consultations with the Senate.

It is appropriate to search a particular category when its elements might have some contextual relationship to the item being searched for. Thus, categories must have associated with them knowledge about how they relate to other categories in memory. The "diplomatic meetings" category, for example, must specify that its events are normally part of "negotiations".

6.3 Memory search

Information relating categories to each other can be used to guide search key construction and memory search. A category should be searched when it is contextually related to the item being sought. Thus, if a negotiations category has associated with it that negotiations are often initiated by meetings between heads of state, and that their sequence of events normally includes a number of diplomatic meetings, then searches for negotiating episodes can include searches for meetings with appropriate heads of state. If it also has associated with the negotiations category the fact that diplomatic trips normally co-occur with negotiations, then it can search for diplomatic trips to appropriate places to find negotiations episodes.

In addition to the domain-specific knowledge necessary for guiding search key construction and memory search, more general search mechanisms or strategies must be used to guide application of that knowledge. Consider, for example, a system with the following strategy:

> Construct search keys and search for episodes in which the target item could have been included. When one is found, see if it includes the target item.

With that strategy and the knowledge that meetings are normally part of negotiations, that system would be able to answer the question "When was the last time Vance talked to Gromyko about SALT?" by searching for the most recent negotiating episode involving SALT, and looking in its sequence of events for a meeting between Vance and

Gromyko. Rules such as this one access the domain-specific knowledge associated with memory categories to direct search key construction and memory search.

7.0 Implementation

Together, CYRUS (Kolodner, 1978; Schank and Kolodner, 1979) and FRUMP (DeJong, 1979) comprise a self-updating information retrieval system which can be queried in English -- CyFr. FRUMP reads stories from the United Press International news wire and outputs a conceptual representation of the important events of each story that it understands. Conceptual summaries concerning Cyrus Vance are sent to CYRUS, which fills in contextual details and adds the new information to its data base. CYRUS can then be interrogated about the new events.

7.1 The FRUMP program

FRUMP (Fast Reading Understanding and Memory Program) is a computer system that has been built based on the idea that world knowledge should be allowed to affect the interpretation of words. FRUMP skims input text for important facts rather than reading it for detail. The flexibility and robustness of the approach is demonstrated by FRUMP's ability to correctly process English text which has never before been seen by either the program or its programmers. Furthermore, the domain of the program is not excessively constrained. FRUMP is designed to work on news articles.

The program can process text from diverse domains such as reports of plane crashes, countries establishing diplcmatic ties, forest fires, and wars. New domains are added by supplying only the domain specific world knowledge; FRUMP's linguistic knowledge is independent of any particular domain. These abilities make FRUMP ideal for the task of "understanding" story texts prior to their insertion into a conceptual data base.

A UPI news wire is connected to the Yale computer to provide real world data for FRUMP. Thus, FRUMP processes actual news articles from the UPI news wire and sends the resulting conceptual summaries to CYRUS. FRUMP is also very efficient. It can easily keep up with the rate news stories arrive over the UPI news wire.

FRUMP uses a data structure called a _sketchy script_ to organize its knowledge about the world. Each sketchy script is the repository for the knowledge FRUMP has about what can occur in a given situation. FRUMP currently has sketchy scripts for approximately 60 different situations ranging from earthquakes to countries establishing diplomatic ties to labor strikes.

Scripts have been used before for natural language processing (Schank & Abelson 1977;, Cullingford 1978). However, they were detailed scripts. A detailed script contains all of the events that might occur in a situation; a sketchy script contains only the important events. FRUMP uses its scripts in a much more integrated manner. The system's world knowledge (in the form of sketchy scripts) is applied to disambiguate individual words as well as connecting related events.

In understanding a story, FRUMP goes through two phases. First it must select a sketchy script. Second, it processes the story using the selected sketchy script to predict the conceptual nature of the important events in the story.

FRUMP's scripts are indexed conceptually. That is, conceptualization patterns indicate which script is appropriate. If FRUMP can build a particular conceptualization which matches one of these patterns, the corresponding sketchy script is loaded. For example, the conceptualization pattern depicting a representative of one government going to the territory of another, indicates that the script for processing diplomatic meetings ($MEET) is appropriate. Suppose FRUMP builds the particular conceptualization representing Cyrus Vance going to Israel. This particular conceptualization matches the above general pattern, so the diplomatic meeting script would be loaded to process the remainder of the story. The script identification phase must be completed in the first paragraph of the story or FRUMP will assume it does not possess the correct sketchy script and give up processing the story.

In the second phase, FRUMP processes the input story guided by the sketchy script selected in the first phase. The sketchy script contains conceptual specifications of the important events that are likely in its situation. FRUMP can then actively look for this important information in the news story. In the case of the Israeli troops crossing into Lebanon, FRJMP can immediately resolve the word "troops" to its "military personnel" meaning once the $WAR sketchy script has been loaded.

FRUMP currently has 60 sketchy scripts in its repertoire. They are:

$ACCUSE
$AGREE
$AID
$APPROVAL
$ARREST
$ASSAULT
$ASYLUM
$AWARD
$BLOCKADE
$NEGOTIATE
$BREAK-RELATIONS
$NEWS-BRIEF
$COMMENT
$CRIME
$DEMONSTRATE
$DENY
$DEATH
$DEMAND
$DEPORT
$ELECTION
$EXPEL
$EXPLODE
$FINANCE
$FIND-RESOURCE
$FLOOD
$GRANT
$HOSPITAL
$KIDNAP
$KILL
$MAKE-RELATIONS
$MEET
$MILITARY-MOBILIZATION
$NATIONALIZE
$FAIL-AGREEMENT
$REFUSE-COMMENT
$POST
$PRICE
$PROMISE
$PROPOSE
$PROTEST
$EARTHQUAKE
$REDUCE-AID
$REQUEST
$RIOT
$REJECT-PROPOSAL
$SEIZE
$SEND
$STRUCTURE-FIRE
$OIL-SPILL
$SPORT-GAME
$STORM
$STRIKE
$WEAPONS-TEST
$THEFT
$THREATEN
$URGE
$VEHICLE-ACCIDENT
$VISIT
$WAR
$WARN

FRUMP has a vocabulary of approximately 2300 root words. It also has morphological rules which allow it to recognize regular plurals, past participles (ed en), gerunds (ing) and nominalizations (tion) forms from root words which extends its vocabulary considerably.

The following is an example of FRUMP processing a news article.

```
*************

INPUT:
      A240 R I SS BYL U V CZC
      AM-PAL SKED 8 -18--------BY FERNANDO DEL MUNDO MANILA, PHILIPPINES
 (UPI)-A bomb exploded aboard a Philippine Airlines jetliner at 24,000 feet
 Friday but the only fatality was the bomber, who was sucked out a six-foot
 -wide hole blasted in the wall of the plane's toilet.
      The twin-engine British-built BAC-111 jet landed safely in Manila
 despite loss of pressurization.  Three persons aboard the plane suffered
 minor injuries.
      Officials said Rodolfo Salazar, an electrician from Cebu, 350 miles
 south of Manila, went into the toilet before the blast and was not among
 the 78 passengers and six crewmembers accounted for later.
      "All circumstances point to the fact that he carried the bomb," an
 official said.
      Intelligence agents said the explosive may have been a sister banaag.
 The passengers were held for about four hours for questioning and released.
      ----UPI 8 -18 3:47 PED----***

SELECTED SKETCHY SCRIPT $EXPLODE

DONE - CPU TIME FOR UNDERSTANDING = 8353 MILLISECONDS

  ENGLISH SUMMARY:
      A BOMB EXPLOSION IN A PHILIPPINES AIRLINES JET HAS KILLED THE PERSON
 WHO PLANTED THE BOMB AND INJURED 3 PEOPLE.

  CHINESE SUMMARY:
      I JIAH FEIHARNG PENNSHEHKEHJI SHANQ DE JAHDANN BAWJAH JAHSYYLE
 FANQJYH JAHDANN DE REN ERLCHIEE SHANQLE SAN GE REN.

  SPANISH SUMMARY:
      UNA EXPLOSION DE BOMBA DENTRO DE UN JET DE LA AEROLINIA FILIPINA HA
 MATADO AL BOMARDERO Y HA HERIDO A 3 PERSONAS.

*************
```

The capitalized letters and numbers preceding and following the story are UPI codes. These are ignored by FRUMP except to identify the date and reporting location.

Here FRUMP identified the topic of the article to be a bomb detonation and selected the sketchy script $EXPLODE. The sketchy script specifies what is important for FRUMP to find in stories about explosions. Of these, FRUMP was able to find the location of the bomb, and a specification of who was killed and injured. As the printout indicates, FRUMP took approximately 8.5 seconds of CPU time to process the story on a DEC PDP 20/60 computer. A story of this length takes about a minute to receive from the UPI wire. Thus FRUMP can quite easily process UPI stories in real time.

For a more complete discussion of the FRUMP system see DeJong (1979a) or DeJong (1979b).

7.2 The CYRUS system

CYRUS is a memory model that organizes biographical information about people. It uses knowledge about its organization for retrieval and automatic updating. CYRUS is an implementation of some of the ideas about intelligent memory organization and retrieval described above. Another important aspect of the CYRUS system is that the organization of its memory represents an attempt to model memory in people, and its retrieval and updating procedures mirror the way we believe people access their memories.

CYRUS has two modules -- a question-answering module which answers questions put to it by a human user, and an updating module which automatically adds new information to memory after that information has been pre-processed by FRUMP. The CYRUS system contains information about former U. S. Secretary of State Cyrus Vance, who was chosen as the model for the system since he is in the news often enough to generate a large number of news updates. More recently, CYRUS has begun collecting information about U. S. Secretary of State Edmund Muskie. The initial memory organization CYRUS starts out with for each of these men is the same. However, we expect the more specific new categories built by the system for Muskie´s events to be somewhat different than those it has built for Vance.

Because CYRUS stores episodes, its memory is organized in episodic categories. These categories are called Memory Organization Packets, or MOPs (Schank, 1979). Similar episodes are stored in the same MOP, along with thc generalized knowledge built up from the similarities in the episodes. Thus, MOPs act as event categories in memory holding episodes and knowledge about those episodes. The generalized information a MOP holds includes such things as typical preconditions and enablement conditions for the episodes, the typical sequence of events for episodes of that class, larger episodes they are usually part of, their usual results, typical location, duration, participants, etc.

One of the uses of generalized information is for inference making. If CYRUS is asked a question such as "What did Vance talk to Gromyko about last time he met with him?", and if it doesn´t have more specific information about the particular episode, it can answer "probably SALT". Similarly, when a new meeting between Vance and Gromyko is added to memory, CYRUS infers that the meeting was probably about SALT.

MOPs are specified conceptually. Some of the MOPs CYRUS uses are "diplomatic meetings", "briefings", "diplomatic trips", "speeches", and "negotiations". Diplomatic meetings are recognized as meetings between diplomats of different countries. Thus, the meetings Vance has with Gromyko, Begin, Sadat, etc., are all diplomatic meetings. Briefings are meetings between diplomats of the same country. When Vance meets with Carter or members of Congress, he is attending briefings. Besides organizing episodes, MOPs also organize more specific MOPs. Some of the more specific MOPs for Vance organized by "diplomatic meetings" are "diplomatic meetings with heads of state", "peace talks", "meetings in the Middle East", and "meetings about SALT". Thus, MOPs and their more specific MOPs are organized hierarchically.

7.3 Memory update in CYRUS

When a new story is sent to CYRUS from FRUMP, CYRUS must add the events in that story to its memory. Its first step is to make necessary inferences to decide which MOP the new events fall into. Thus, if FRUMP sends CYRUS a summary such as "Vance and Gromyko met yesterday in Moscow to talk about SALT II", CYRUS infers that the meeting was a diplomatic meeting. It also fills in contextual details based on that initial categorization. For the summary above, it infers that the meeting was part of SALT II negotiations and that Vance must have been on a diplomatic trip.

After making this initial categorization and inferring contextual details, the event is added to the chosen MOP. A MOP organizes more specific MOPs and also indexes events according to their differences and variations from the norm. A new event being added to a MOP is indexed according to each of its differences and variations from the norm. In adding an event to a MOP, then, it will either be indexed into another more specific MOP where the same indexing will happen, it will be indexed uniquely, or it will fall into an index point inhabited by another event. The event above would be indexed as a "meeting with Gromyko", "meeting in Moscow", "meeting about SALT", etc.

If the new event is indexed to a point which already holds an event, the system looks at the similarities and differences between those events, makes generalizations based on that, and indexes the two events according to their differences. In that way, new more specific memory categories or MOPs are formed. There are two important issues which must be addressed in doing the indexing which organizes memory. First of all, the norms for each category must be available so that differences can be picked out. Second, the right differences must be focused on. One of the specifications a MOP in CYRUS has is its norms, or the similarities between its members. CYRUS starts out with knowledge about the norms for meetings, trips, etc., and as it builds new categories from those, it generalizes the norms for those categories. CYRUS also starts out with knowledge in each of its domains about what to focus on for indexing. It knows that the topics of meetings are important, that the destination and goal of a trip are important, etc. It knows that the occupation and nationality of participants are important for diplomatic events, and for all events, it knows that the participants, results, sequence of events, larger episodes, etc. are important to index on. It does not index on those things, however, when they conform to the values predicted for events of that type.

In the following example, CYRUS is adding a new meeting with the Israeli defense minister to its memory. The MOP it is adding that event to is the $MEET MOP, the MOP for diplomatic meetings. Because that meeting is so similar to another meeting it already has in its memory, it is "reminded" of that meeting, automatically creates a new MOP for meetings of that type, and makes generalizations about other aspects of those meetings.

```
*************

Adding     $MEET actor (Vance)
                 others (defense minister of Israel)
                 topic (Military aid to Israel)
                 place (Jerusalem)
to memory ...

Reminded of $MEET actor (Vance)
                  others (defense minister of Pakistan)
                  topic (Military aid to Pakistan)
                  place (Washington)
because both are "diplomatic meetings"
        both have contract topic "military aid"
    creating new MOP:  diplomatic meetings about military aid
  generalizing that when
    Vance meets about military aid,
    often he meets with a defense minister

*************
```

The similarities CYRUS notices are directed by the domain. It is important to notice the occupation of participants in an occupational episode, and important to notice the topic of a meeting. Although it holds the information that both Jerusalem and Washington are cities, it did not notice that similarity. Nor did it notice that both defense ministers were male. Neither of those observations are important to meetings, and in fact both are the norm.

CYRUS does not create new MOPs for every event it adds to memory. The following event is a meeting with Begin about the Camp David Accords. There were already a number of other meetings in memory similar to this one when it was added, and it had no aspects different from those other meetings. Thus it was added to MOPs that were already in memory.

```
*************

Adding     $MEET actor (Vance)
                 others (Begin)
                 topic (Camp David Accords)
                 place (Jerusalem)
to memory ...

Putting it into MOP: meetings with Begin in Israel
    confirming generalizations
Putting it into MOP: meetings about the Camp David Accords with
                        Israeli participants
    confirming generalizations
Putting it into MOP: meetings in Israel
    confirming generalizations
       ...

*************
```

7.4 Retrieval in CYRUS

CYRUS´ other module takes care of retrieval. Organization according to differences allows CYRUS to retrieve events that differ from the norm more easily than those that are typical. It also makes it possible to answer questions about typical activities without having to retrieve all or many distinct episodes. When a question is asked, CYRUS analyzes it to see which MOPs its answer should be found in. The answer to the question "Has Vance met with Gromyko recently?" would be found in a "diplomatic meetings" MOP, while the answer to "Has Vance met with Carter recently?" would be found in a "briefings"

MOP. Rules similar to those used to categorize events are used to categorize questions. These rules are called "context construction rules". When a category has been selected, the description of the event in the question is indexed into the appropriate MOP hierarchy as far as it will go, in the same way events are added to memory. If it ends at an index point with only a few events, it extracts those that match, and uses them to answer the question. The question "Has Vance met with Gromyko recently?" thus might find the event "Vance had a meeting with Gromyko about SALT yesterday". To answer, "When was the last time Vance talked to Gromyko?", CYRUS first uses context construction rules to infer that they would have talked at a diplomatic meeting, then searches the MOP hierarchy for diplomatic meetings to find the one that fits the bill.

```
*************

>When was the last time Vance talked to Gromyko?

The question concept is:
((ACTOR HUM1 <=> (*MTRANS*) TO HUM66) TIME TIM37)

 inferring a diplomatic meeting
searching memory for $MEET with participants = Gromyko
   found (CON432)
answering question using time context

The answer is:
   On October 22 in Russia.

*************
```

Because descriptions of events in questions are usually not as elaborate those of actual events, it is not always possible to extract a unique event from memory. In that case, search strategies are applied to retrieve possible related contexts which might point to the event in question. Thus, one way to find a recent meeting between Vance and Gromyko is to search for recent SALT negotiations (which don't happen as often and might be easier to find), and if it is found, search its sequence of events. CYRUS makes use of approximately 10 search strategies based on those we observed people using to answer questions. If a unique event still cannot be found, generalized information can be used to infer a probable answer. Thus, to enumerate all recent meetings Vance has had with Gromyko, CYRUS first attempts to retrieve recent meetings, then applies search strategies to search for recent negotiating episodes with Gromyko, recent conferences Gromyko attended, and recent diplomatic trips to Russia.

```
*************

>How many times has Vance talked to Gromyko recently?

The question is:
((ACTOR HUM1 <=> (*MTRANS*) TO HUM66) TIME TIM31 QUANTITY (*ALL*))

   inferring a diplomatic meeting
searching memory for $MEET with participants = Gromyko
   found (CON1338 CON1354 CON1362)
applying strategies to search memory
  checking locational information on input
    meeting could have occured during diplomatic trip to the USSR
    searching for I-VIPVISIT to the USSR
     found (CON1853 CON1314)
      searching I-VIPVISITs for meetings
        found (CON1338 CON1354 CON1362)
  collecting simple MOPs input could have occured in --
     (sM-SUMMIT-CONFERENCE sM-CONFERENCE)
     searching for sM-SUMMIT CONFERENCE with participants = Gromyko
        didn't find any
     searching for sM-CONFERENCE with participants = Gromyko
        didn't find any
  collecting IMOPs input could have occured in --
   (I-NEGOTIATE)
     searching for I-NEGOTIATE with participants = Gromyko
        found (CON1649)
      searching I-NEGOTIATE instance for input
        found (CON1751 CON1771 CON1784)

The answer is:
(CON1784 CON1771 CON1751 CON1362 CON1354 CON1338)

   At least six times in the last 4 months.

*************
```

For more information about CYRUS, see Kolodner (1978, 1980) and Schank and Kolodner (1979).

7.5 The complete system

In CyFr's normal operation, FRUMP reads stories from the UPI wire and sends representations of stories about Vance and Muskie to CYRUS. When CYRUS receives a story representation from FRUMP, it updates its memory, as described above. This consists of adding the new information to the correct categories, making generalizations about the contents of those categories, and creating new categories as necessary. CYRUS can then answer questions posed in English about the new data. Although individually CYRUS and FRUMP have been in operation for over two years, they have only been combined recently. At this writing, CyFr has processed approximately 50 stories about Vance and approximately 10 more about Muskie.

Of course, there are many ways the system can be extended. The types of activities CyFr is interested in might be extended so that it can process news items about a broader range of topics than just activities of diplomats. Also we expect to increase the communication between FRUMP and CYRUS. Currently the communication between the two programs is unidirectional -- from FRUMP to CYRUS. We hope to extend that connection in two ways. First, we would like CYRUS to send messages to FRUMP telling FRUMP what it expects to see and would like to find out from a story. This information could be used to help guide FRUMP's text processing. Second, we would like FRUMP to be able to request information from CYRUS' memory. Sometimes it is necessary to know what has happened earlier in order to understand a current story. Extending the systems as just described will create a more integrated system which should add to the power, flexibility and efficiency of processing.

```
*****************

@RU LFRUMP

Dictionary loaded
Building script trees
Dictionary fork...Done

*(SKIM (MAY130.K05]

FILE (MAY130 . K05) SKIMMED AT 5:31PM ON 5-14-1980

INPUT:  a074 r i ss czc bsa u v
     -Skie 1 d-picp6thraf 5-13-------------- (Me lves foreure (
     ---By Jim Anderson---Washington (Upi)-Secretary of State Edmund Muskie
flew to Europe on his maiden diplomatic mission today to whip up allied
support for U.  S.  policies toward Iran and to meet with veteran Soviet
Foreign Minister Andrei Gromyko.
     Muskie met for 20 minutes with President carter, then left nearby
andrews Air Force Base for Brussels shortly after 9 a.  m.  Edt.
     Muskie will meet with NATO foreign ministers in Brussels Wednesday,
then go to Vienna for a meeting with Gromyko at the 25 th anniversary
celebration of the Austrian State Treaty.
     Monday, Muskie told a group of business executives:  "There has been a
perception that the allies are less than enthusiastic" about the program of
U.  S.  economic sanctions against Iran.  mandkow i am ous abutt, "esai.
pcup6ta:  kesd Ao------------------5-13 9:41 a----kkk

SELECTED SKETCHY SCRIPT $MEET

CPU TIME FOR UNDERSTANDING = 87463 MILLISECONDS SKIM done

** Generating ...

ENGLISH:
     Secretary of State Edmund Muskie went on a plane from the United
States to Europe today.  Secretary of State Edmund Muskie had talks with
Carter during a briefing.  Secretary of State Edmund Muskie gave a speech
to a group on Monday.  Secretary of State Edmund Muskie will have talks
with NATO in Brussels on Wednesday.

sending summary to CYRUS -- file ((CYRUS FRUMP-DUMPS) (MAY130.R05))

******************

@cyrecv

*(PROCESS)

reading in file ((CYRUS FRUMP-DUMPS) (MAY130 . R05)) from FRUMP
converting file to CYRUS format ...

Conversion complete.
Sending file ((CYRUS FRUMP-DUMPS) (MAY130 . F05)) to CYRUS
```

```
******************

@CYRUS

today is Friday, May 16, 1980.

*(PROCESS-FILES)

reading in file (MAY130 . F05)
updating memory with new events ...
updating complete

*******************

>Where is Muskie today?
  ...searching memory for question concept
Probably in the United States.

>Where was Muskie Tuesday?
   ...searching memory for question concept
In Europe.

>Why did he go there?
   ...answering question using previous context
To meet with Gromyko.

>Who else did he meet with there?
   ...answering question using previous context:
With NATO in Brussels on Wednesday.

>When was the last time Muskie made a speech?
  ...answering question using time context
On Monday, May 12.

*******************
```

REFERENCES

Allen, L. E. (1966). A Language-normalization approach to information retrieval in law. Presented at the 1966 annual meeting of The American Political Science Association.

Birnbaum, L. and Selfridge, M. (1979). Problems in conceptual analysis of natural language. Research Report #168. Department of Computer Science. Yale University, New Haven, CT.

Bobrow, D. G. and Winograd, T. (1977) An overview of KRL a knowledge representation language. Cognitive Science, Vol. 1, pp. 3-46.

Charniak, E. (1977) A framed painting: the representation of a common sense knowledge fragment. Cognitive Science, Vol. 1, pp. 355-394.

Cullingford, R. (1978). Script application: computer understanding of newspaper stories. Ph. D. thesis. Research Report #116. Department of Computer Science. Yale University, New Haven, CT.

DeJong, G. F. (1979a). Skimming stories in real time: An experiment in integrated understanding. Research Report #158. Department of Computer Science. Yale University, New Haven, CT.

DeJong, G. F. (1979b). Prediction and substantiation: a new approach to natural lnguage processing. Cognitive Science, Vol. 3, pp. 251-273.

Grishman, R. and Hirschman, L. (1978). Question answering from natural language medical data bases. Artificial Intelligence, Vol. 11, pp. 25-43.

Heaps, H. S. (1978). Information Retrieval: Computational and Theoretical Aspects. Academic Press, New York.

Hendrix, G. G., Sacerdoti, E. D., Sagalowicz, D., and Slocum, J. (1978). Developing a natural language interface to complex data. ACM Transactions on Database Systems, Vol 3, No. 2., pp. 105-147.

Kolodner, J. L. (1978). Memory organization for natural language database inquiry. Research Report #142. Department of Computer Science. Yale University, New Haven, CT.

Kolodner, J. L. (1980). Organization and Retrieval from Long Term Episodic Memory. Ph.D. Thesis (forthcoming). Department of Computer Science, Yale University.

Lebowitz, M. (1979) Reading with a purpose. Proceedings of 17th Annual Meeting of the Association of Computational Linguistics, San Diego, CA.

Norman, D. E., Rumelhart, D. E., & the LNR Research Group (1975) Explorations in cognition, Freeman, San Fransisco.

Riesbeck, C. and Schank, R. C. (1976). Comprehension by Computer: Expectation-Based Analysis of Sentences in Context. Research Report 78. Department of Computer Science. Yale University, New Haven, CT.

Salton, G. (1978). Automatic Content Analysis in Information Retrieval. Technical Report #68-5. Department of Computer Science. Cornell University, Ithica, New York.

Schank, R. C. (1975). Conceptual Information Processing. North Holland, Amsterdam.

Schank, R. C. (1979). Reminding and memory organization: An introduction to MOPs. Research Report #170. Department of Computer Science, Yale University.

Schank, R. C. and Abelson, R. P. (1977). Scripts, Plans, Goals, and Understanding. Lawrence Erlbaum Press, Hillsdale, N.J.

Schank,R. C., and Carbonell, J. (1979). Re: The Gettysberg Address: Representing Social and Political Acts. In Findler, N. (ed). Associative Networks: Representation and Use of Knowledge by Computers. Academic Press, New York. pp. 327-362.

Schank, R. C. and Kolodner, J. L. (1979). Retrieving Information from an Episodic Memory Research Report #159. Department of Computer Science, Yale University. Short version in Proceedings of the Sixth International Joint Conference on Artificial Intelligence, Tokyo.

Waltz, D. L. (1978). An English language question answering system for a large relational database. Communications of the ACM, Vol. 21, No. 7, pp. 526-539.

Wilensky,R. (1978). Understanding goal-based stories. Ph. D. Thesis. Research Report #140. Department of Computer Science. Yale University, New Haven, CT.

Wilks, Y. (1977) Knowledge structures and language boundries. Proceedings of the Fifth International Joint Conference on Artificial Intelligence, Cambridge, MA.

Woods, W., Kaplan, R., and Nash-Webber, B. (1972). The LUNAR sciences natural language information system: final report. Bolt, Beranek and Newman Report No. #2378, Cambridge, MA.

DATE
ILMED
8